# Físréal: A Low Cost Terabyte Search Engine

Paul Ferguson, Cathal Gurrin, Peter Wilkins, Alan F. Smeaton,

Centre For Digital Video Processing,
Dublin City University,
Ireland.
{paul.ferguson, cathal.gurrin, peter.wilkins and alan.smeaton}@computing.dcu.ie

**Abstract.** In this poster we describe the development of a distributed search engine, referred to as Físréal, which utilises inexpensive workstations, yet attains fast retrieval performance for Terabyte-sized collections. We also discuss the process of leveraging additional meaning from the structure of HTML, as well as the use of anchor text documents to increase retrieval performance.

## Introduction

As the size of the web increases, the task of developing a cost-effective search engine to deal with these large amounts of documents becomes a major engineering task. The presence of a new Terabyte track in TREC2004 is just one example of how important large-scale retrieval has become. The test collection used for the Terabyte track was the GOV2 collection, which consists of a large portion of the .GOV domain (25,205,179 documents). In this poster we describe an architecture for a distributed search engine which we have used to provide fast search facilities over collections including GOV2 and the even larger (94.5 million document) SPIRIT collection.

## Retrieval Architecture

Físréal was designed to be a scalable, distributed search service. Our solution for this was to distribute the index across several machines and to provide a search engine or leaf server on each of these machines. These leaf nodes receive their queries from an aggregate engine which receives the initial query and distributes the query to each of the node engines, then combines the results from each before presenting a final ranked list. The hardware we employed consisted of DELL PowerEdge 600SC Servers, each with, one P4 2.4 Ghz CPU, 2GB RAM, 4x250GB IDE hard drives with RAID 0. The approximate cost of each server (Nov 2004) is €2,300.

One example implementation of Físréal (for the SPIRIT collection) required four leaf servers, with the collection being split arbitrarily into equal portions for each server. Experience gained in indexing SPIRIT suggested that retrieval performance

was related to the number of leaf servers implemented, therefore when indexing the GOV2 collection (substantially smaller) we also employed four leaf servers.
*Aggregate Server:* The aggregate server provides the interface to the search service and handles all communication with the leaf servers. The aggregate itself does not directly reference any index.

*Leaf Server:* A leaf server is a single instance of the search engine and could be independently queried. As many leaf servers as required can be employed to produce the distributed search engine. Each of the leaf servers receives the same query from the aggregate server. A global lexicon and total number of occurrences for each term in the entire collection is held at each leaf server so that the correct ranking score for each document can be calculated as it would have been done if the entire index had been held on a single machine.

## Indexing Issues

The following index structures are currently supported by the search engine:

*Standard Index:* The index employed by each leaf server is similar to a conventional inverted index. Essentially, for each term in a collection-wide lexicon, there is an object that contains the list of documents where that term occurs, and its corresponding TF. This object is sorted by the TF for each term normalised by the document length so as to allow the retrieval of the top subset of documents associated with each term. The effect of this on retrieval performance is currently under investigation and we will soon report concrete findings of the trade-off between precision, the proportion of the index examined and performance.

*Weighted Index:* It is believed that certain HTML tags contain text that is more representative of the content of the document than other text in the document. For example, text in the title tag would generally be more reflective of the content of the document than the body text of the document, and should therefore be given more weighting in retrieval. To infer these weightings, we defined what we considered to be the tags that would contain the most representative words, then we defined the extra weighting for terms in these tags based on previous work in TREC and elsewhere. These tags and their weightings are as follows:

| **Tag** | TITLE | B | H1 | H2 | H3 | H4 | H5 | H6 | I | EM | U | A | ALT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Weight** | 6 | 4 | 4 | 4 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 4 | 6 |

These tags were identified in each document at indexing time, and each embedded term was given the appropriate weighting to be incorporated into the index during the indexing stage. The structure of the weighted index took on a similar structure to the conventional index, the difference being that along with the TF of the term the weighted TF was also held for each term in each document that it occurred in. This allows the same index to support either a weighted or non-weighted ranking for each query as specified at query time yet has a negligible effect on query processing time.

This index has an average size of 12.6GB per leaf node, where each index is used to index over 23 million documents.

*Anchor Text Index:* It is believed that integrating anchor text into the retrieval process can be useful in improving retrieval performance for Web IR[1]. We generated anchor text surrogate documents by extracting the anchor text (with a window of 50 bytes either side, to the nearest word) from all documents that link to a given document. This creates a collection of documents which contains terms that the in-link authors used to describe the target document. We created an index from these documents with the same structure as the conventional index. This index has an average size of 1.86GB per leaf node

*URL Index:* An index consisting of terms obtained by breaking up a document URLs into terms based on its domain and path. This index has a total size of 2.2GB.

## Retrieval Performance and Conclusion

In order to examine the performance of Físréal we present results from experiments with the GOV2 collections and relevance judgements from the TREC 2004 TB track:

- A baseline BM25 run (BM25: k1=1.2, k3=1000, b=0.75).
- A BM25 run, using a weighted index (WBM25: k1=1.2, k3=1000, b=0.75).
- A run incorporating BM25 (k1=1.2, k3=1000, b=0.75), anchor text (k1=50, k3=1000, b=0), descriptions and URL text (k1=1.2, k3=1000, b=0.75).

All of our runs take only the title of the topic and use this as the query for all fifty queries. The following table presents the performance figures for these experiments.

| RUN | MAP | Recall (10617 relevant) | Top 20 retrieval time (Note: same top 20 as top 10000) |
|---|---|---|---|
| BM25 | 0.1272 | 6765 (64%) | 1.823 seconds |
| Weighted BM25 | 0.1022 | 6284 (59%) | 1.922 seconds |
| BM_Anchor_URL | 0.1150 | 6765 (64%) | 2.01 seconds |

We have presented an architecture and preliminary results for a low cost distributed search engine. For a five-server implementation the cost is approximately €11,500, and for the GOV2 collection this equates to €450 per million documents indexed. As can be observed from the preliminary results further thought and refinement will need to be given to the weights selected for weighted BM25. We also plan to explore the cost-performance ratio for a larger number of leaf servers.

**Acknowledgement:** This work was supported by Science Foundation Ireland, under grant number 03/IN.3/I361.

## References

1. *N. Craswell and D. Hawking. (2003). Overview of the TREC 2003 Web Track, Proceedings of the Twelfth Text Retrieval Conference, pg78-92, 2003.*